AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings of claims in the application. Applicants have submitted a new complete claim set showing marked up claims with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing.

1-16. (Canceled)

17. (Currently Amended) ~~The method of claim 16, wherein decreasing progress estimations are prevented~~

A method of estimating query progress, comprising:

receiving a query;

based on characteristics of the received query, determining a model of work to be performed during execution of the query;

estimating a total amount of work that will be performed according to the model;

iteratively estimating an amount of work performed according to the model at a given point during the execution of the query;

iteratively estimating the progress of the query using the estimated amount of work performed and the estimated total amount of work;

iteratively displaying estimated progress of the query to a user; and

preventing decreasing progress estimations from being displayed to the user by using an upper bound on the total work that will be performed as an estimate of the total work that will be performed.

18. (Currently Amended)   The method of claim 1<u>7</u> further comprising identifying a spill of tuples during query execution and adjusting the model of work to account for additional work that results from the spill of tuples.

19-20. (Canceled)

21. (Currently Amended)   The method of claim 1<u>7</u> further comprising updating an estimated total amount of work that will be performed during query execution.

22. (Currently Amended)   The method of claim 1<u>7</u> wherein an estimated amount of work performed according to the model is updated at a plurality of points during query execution.

23. (Currently amended)   ~~The method of claim 1 further comprising~~
<u>A method of estimating query progress, comprising:</u>
<u>receiving a query;</u>
<u>based on characteristics of the received query, determining a model of work to be performed during execution of the query;</u>
<u>estimating a total amount of work that will be performed according to the model;</u>
<u>iteratively estimating an amount of work performed according to the model at a given point during the execution of the query;</u>
<u>iteratively estimating the progress of the query using the estimated amount of work performed and the estimated total amount of work;</u>
<u>iteratively displaying estimated progress of the query to a user;</u>

preventing decreasing progress estimations from being displayed to the user; and

maintaining an upper bound and a lower bound on the on the total amount of work that will be performed and modifying an estimated total amount of work that will be performed when the estimated total amount of work that will be performed is outside a range defined by the upper bound and the lower bound.

24. (Currently amended)   The method of claim 3, ~~further comprising maintaining an upper bound and a lower bound on the on the total amount of work that will be performed and~~ wherein the amount of work performed during execution of a query is modeled as a number of items returned by a query operator, and further comprising modifying an estimated total number of items that will be returned by the query operator when the estimated total number of items that will be returned by the query operator is outside a range defined by the upper bound and the lower bound.

25. (Original) The method of claim 24 wherein a rule used for maintaining a bound on the total number of items that will be returned by the query operator is specific to the query operator.

26. (Original) The method of claim 25 wherein the query operator is a Group By operator and the rule used for maintaining an upper bound on a number of groups that will be returned by the Group By operator comprises subtracting a number of items returned by an immediately preceding operator in a query execution plan from an upper bound of the immediately preceding

operator and adding a number of distinct values observed by the Group By operator.

27. (Original) The method of claim 25 wherein the query operator is a Hash Join operator and the rule used for maintaining an upper bound on the number of rows that will be returned by the Hash Join operator comprises subtracting a number of items returned by an immediately preceding operator in a query execution plan from an upper bound of the immediately preceding operator and multiplying a number of rows of a largest build partition.

28. (Original) The method of claim 24 further comprising setting the lower bound to a number of items returned by the query operator at a given point during query execution.

29. (Original) The method of claim 24 wherein the upper bound of the query operator is maintained using an upper bound of one or more preceding query operators in a query execution plan.

30. (Original) The method of claim 29 wherein the upper bound of the query operator is maintained using an upper bound of an immediately preceding query operator in the query execution plan.

31. (Original) The method of claim 24 wherein the upper bound of the query operator is maintained using a number of items returned by one or more preceding operators in a query execution plan at a given point during query execution.

32. (Original) The method of claim 31 wherein the upper bound of the query operator is maintained using a number of items returned by an immediately preceding query operator in the query execution plan.

33. (Original) The method of claim 24 wherein the upper bound of the query operator is maintained using a number of items returned by the query operator at a given point during query execution.

34. (Original) The method of claim 24 wherein upper and lower bounds are maintained for a plurality of query operators in a query execution plan and wherein a changes in bounds of query operators are periodically propagated to other query operators in the query execution plan.

35. (Currently amended)   Computer readable media comprising computer-executable instructions for performing the method of claim 1_7_.

36-43. (Canceled)

44. (Currently amended)   ~~The user interface of claim 36 further comprising~~

In a computer system including a display, a user input facility, and an application for presenting a user interface on the display, a user interface comprising:

a query progress indicator that estimates progress based on:

an estimated amount of work performed at a given point during execution of the query according to an execution plan generated in response to a received query;

an estimated total amount of work to be performed according to the execution plan, wherein the execution plan defines work to be performed during execution of the query;

a query end selector that allows the user to abort execution of the query; and

a tuple spill indicator that alerts a user when tuples spill to disk during query execution.

45– 46. (Canceled)

47. (Currently amended) ~~The system of claim 45 wherein the processor identifies~~

A system for providing an indication of query progress, comprising:

a user input device enabling a user to begin execution of a query and abort execution of a query;

a display;

a data content that queries can be executed upon;

a memory in which machine instructions are stored;

a processor that is coupled to the user input device, to the display, to the data content, and to the memory, the processor executing the machine instructions to carry out a plurality of functions, including:

receiving a query;

generating an execution plan in response to the received query;

selecting a model of work corresponding to the execution plan;

executing the query upon the data content;

estimating progress of the query using the selected model of work; and

<u>providing an indicator of query progress on the display;</u>

<u>identifying</u> a spill of tuples during query execution and provides an indication of the spill on the display.

48. (Currently amended)   The system of claim [[45]]<u>47</u> the indicator of query progress provides a visual indication of a percentage of query execution that has been completed.

49. (New)   The method of claim 17 wherein work performed during execution of a query is modeled as a number of items returned by a query operator.

50. (New)   The method of claim 17 wherein work performed during execution of a query is modeled as a number of GetNext() calls by a query operator.

51. (New)   The method of claim 17 wherein the work performed during execution of the query is modeled as work performed by a driver node operator during execution of the query.

52. (New)   The method of claim 17 wherein work performed by a driver node operator is modeled as a number of items returned by the driver node operator.

53. (New)   The method of claim 17 wherein work performed by a driver node operator is modeled as a number of GetNext() calls by a driver node operator.

54. (New)   The method of claim 17 further comprising dividing a query execution plan into a set of pipelines and estimating the progress of each pipeline.

55. (New)   The method of claim 54 wherein the pipelines comprise sequences of non-blocking operators.

56. (New)   The method of claim 54 further comprising combining progress estimates for the pipelines to estimate the progress of the query.

57. (New)   The method of claim 56 further comprising assigning weights to the pipelines.

58. (New)   The method of claim 57 wherein the weights are based on relative execution rates of the pipelines.

59. (New)   The method of claim 54 further comprising initializing an estimate of the total amount of work that will be performed by a pipeline with an estimate from a query optimizer.

60. (New)   The method of claim 17 further comprising refining the initial estimate of the total work using feedback obtained during query execution.

61. (New)   The method of claim 54 further comprising identifying driver node operators of the pipeline and modeling the work performed during execution of the pipelines as work performed by the driver node operators.

62. (New)   The method of claim 54 further comprising modeling the work performed during execution of the pipelines as work performed by all operators in the pipeline.

63. (New)   The method of claim 54 further comprising identifying driver node operators of the pipeline and using information about the driver node operators obtained during execution to estimate a total amount of work that will be performed by all operators in the pipeline.

64. (New) The method of claim 17 further comprising preventing decreasing progress estimations from being displayed to the user.

65. (New)   The user interface of claim 44 wherein the query progress indicator provides a visual indication of a percentage of query execution that has been completed.

66. (New)   The user interface of claim 65 wherein the percentage of query execution that has been completed is estimated by dividing a number of tuples returned by the query by an estimated total number of tuples to be returned by the query.

67. (New)   The user interface of claim 65 wherein the percentage of query execution that has been completed is estimated by dividing a number of

tuples returned by an operator by an estimated total number of tuples to be returned by the operator.

68. (New)   The user interface of claim 65 wherein the percentage of query execution that has been completed is estimated by dividing a GetNext() calls by a query operator by an estimated total number of GetNext() calls by the operator.

69. (New)   The user interface of claim 65 further comprising initializing the estimated total number of GetNext() calls with an estimate from a query optimizer.

70. (New)   The user interface of claim 69 wherein initial estimate of the total number of GetNext() calls is updated using feedback obtained during query execution.

71. (New)   The user interface of claim 44 the query progress indicator is prevented from providing an indication of decreasing query progress.